Machine Learning Project for an online music store that recommends music albums , that users are likely to buy

Importing the data:

In [10]:

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import joblib

data = pd.read_csv("C:\\Users\\Gudise Dharani\\Desktop\\music.csv")
```

Data cleaning:

- It's just a demo sample for this project, yet it's capable enough to work for large dataset using this program.
- There is nothing much to clean in this dataset.

Split into input and output sets:

In [11]:

```
input_set = data.drop(columns = ['genre'])
input_set
```

Out[11]:

	age	gender
0	20	1
1	23	1
2	25	1
3	26	1
4	29	1
5	30	1
6	31	1
7	33	1
8	37	1
9	20	0
10	21	0
11	25	0
12	26	0
13	27	0
14	30	0
15	31	0
16	34	0
17	35	0

In [12]:

```
output_set = data.drop(columns = ['age','gender'])
output_set
```

Out[12]:

	genre
0	HipHop
1	HipHop
2	HipHop
3	Jazz
4	Jazz
5	Jazz
6	Classical
7	Classical
8	Classical
9	Dance
10	Dance
11	Dance
12	Acoustic
13	Acoustic
14	Acoustic
15	Classical
16	Classical
17	Classical

Split into training and testing sets:

In [13]:

```
input_train, input_test, output_train, output_test = train_test_split(input_set, output_set, test_size = 0.2)
```

Building the model:

In [14]:

```
model = DecisionTreeClassifier()
```

Training the model:

In [15]:

```
model.fit(input_train, output_train)
```

Out[15]:

```
DecisionTreeClassifier()
```

Testing the model and evaluating the accuracy:

In [19]:

```
predictions = model.predict(input_test)
score = accuracy_score(output_test, predictions)
score
```

Out[19]:

```
1.0
```

Persisting models:

saving model into file

In [20]:

```
joblib.dump(model, "music_recommender.joblib")
```

Out[20]:

```
['music_recommender.joblib']
```

loading model from file and asking it to make predictions

In [21]:

```
model = joblib.load("music_recommender.joblib")
predictions = model.predict(input_test)
score = accuracy_score(output_test, predictions)
score
```

Out[21]:

```
1.0
```

In [ ]: